CLAIMS

What is Claimed is:

1. A method comprising:

dividing a data word into data sub-words onto sub-word paths;

allowing communication between the sub-word paths; and

encoding the data sub-words into encoded data sub-words;

such that an information content of the data word is spread between the encoded data sub-words and the weight of the encoded data sub-words.

2. A method comprising:

allowing communication between sub-word paths; and

decoding encoded data sub-words into data sub-words;

such that the data sub-words form a data word whereby an information content of the data word is concentrated back into the data word.

3. An encoder module, to encode a data word, whose data elements occupy at least a first logic state and a second logic state, said encoder module comprising:

at least two sub-word paths, each of said at least two sub-word paths to receive a data sub-word, comprising a set of data elements of the data word; and

an encoder coupled with said at least two sub-word paths, said encoder to encode the data sub-word into an encoded data sub-word;

such that encoded data sub-words form an encoded data word wherein an information content of the data word is spread between the encoded data sub-words and the weight of the encoded data sub-words.

4. An encoder module, as in claim 3, wherein said encoder module allows communication of a group of data elements from the data word, such that information is shared between said at least two sub-word paths.

5. An encoder module, as in claim 4, wherein the information is a data sub-word weight.

6. An encoder module, as in claim 4, wherein the information is at least one data element from the data sub-word.

7. An encoder module, as in claim 4, wherein the information is at least one data element from the data word.

8. An encoder as recited in claim 4, wherein said encoder inverts all the data elements, within the data sub-word, whose significance is less than or equal to the significance of a particular data element within the data sub-word.

9. An encoder as recited in claim 4, wherein said encoder inverts all encoded data elements, within the encoded data sub-word, whose significance is less than or equal to the significance of a particular encoded data element within the encoded data sub-word.

10. An encoder as recited in claim 4, wherein the number of elements in the encoded data sub-word is at least one greater than the number of elements in the data sub-word.

11. An encoder module, as in claim 4, further comprising encoding logic for encoding values represented by the data words that are not encoded by said encoder.

12. An encoder module, as in claim 4, further comprising a code governor, wherein said code governor sets a logic state of at least one particular data element location in the encoded data word such that the number of data elements in each logic state is constant for each encoded data word.

13. A code governor, as recited in claim 12, wherein said code governor, uses in part, the information shared between said at least two sub-word paths to set the logic state of at least one particular data element in the encoded data word.

14. An encoder module, as in claim 12, further comprising a parity line to change the weight of the encoded data word by adding an element to the encoded data word.

15. A parity line, as recited in claim 14, wherein said parity line, uses in part, the information shared between said at least two sub-word paths to set the logic state of the element output onto said parity line.

16. An encoder as recited in claim 4, wherein said encoder encodes the data word by associating the data word with an encoded data word which is selected from a set of encoded data words where the encoded data word is of length z elements with a fixed number of data elements in each logic state.

17. An encoder as recited in claim 4, wherein said encoder encodes the data sub-word by associating the data sub-word with the encoded data sub-word in a user defined way.

18. An encoder as recited in claim 4, wherein said encoder is a binomial encoder.

19. An encoder module as recited in claim 4, wherein said encoder module is disposed on an integrated circuit die.

20. An encoder module, as in claim 4, further comprising a *Vdd* power rail wherein a sum of the current flowing in said *Vdd* power rail is substantially constant.

21. An encoder module, as in claim 4, further comprising:

a code governor; and

a *Vdd* power rail;

wherein said code governor sets the logic state of at least one particular data element location in the encoded data word, such that a sum of the current flowing in said *Vdd* power rail is substantially constant.

22. A code governor, as recited in claim 21, wherein said code governor, uses in part, the information shared between said at least two sub-word paths to set a logic state of at least one particular data element in the encoded data word.

23. An encoder module, as in claim 21, further comprising a parity line to change the weight of the encoded data word by adding an element to the encoded data word.

24. A parity line, as recited in claim 23, wherein the information shared between the data sub-words is used, in part, to set a logic state of the element output onto said parity line.

25. An encoder module, as in claim 4, further comprising a *Vss* power rail wherein a sum of a current flowing in said *Vss* power rail is substantially constant.

26. An encoder module, as in claim 4, further comprising:

a code governor; and

a *Vss* power rail;

wherein said code governor sets the logic state of at least one particular data element location in the encoded data word, such that a sum of the current flowing in said *Vss* power rail is substantially constant.

27. A code governor, as recited in claim 26, wherein said code governor, uses in part, the information shared between said at least two sub-word paths to set a logic state of at least one particular data element in the encoded data word.

28. An encoder module, as in claim 26, further comprising a parity line to change the weight of the encoded data word by adding an element to the encoded data word.

29. A parity line, as recited in claim 28, wherein the information shared between the data sub-words is used, in part, to set a logic state of the element output onto said parity line.

30. An encoder module, as in claim 4, further comprising encoded data lines wherein a sum of the current flowing, in said encoded data lines, is substantially constant.

31. An encoder module, as in claim 4, further comprising:

a code governor, wherein said code governor sets the logic state of at least one particular data element location in the encoded data word; and

encoded data lines, connected with said encoder;

such that a sum of the current flowing in said encoded data lines, connected with said encoder is substantially constant.

32. A code governor, as recited in claim 31, wherein said code governor, uses in part, the information shared between said at least two sub-word paths to set a logic state of at least one particular data element in the encoded data word.

33. An encoder module, as in claim 31, further comprising a parity line to change the weight of the encoded data word by adding an element to the encoded data word.

34. A parity line, as recited in claim 33, wherein the information shared between the data sub-words is used, in part, to set a logic state of the element output onto said parity line.

35. A method of encoding a data word, whose data elements occupy at least a first logic state and a second logic state, said method comprising:

receiving data sub-words onto sub-word paths, the data sub-words comprising sets of data elements of the data word; and

encoding the data sub-words into encoded data sub-words;

such that the encoded data sub-words form an encoded data word wherein an information content of the data word is spread between the encoded data sub-words and the weight of the encoded data sub-words.

36. A method, as in claim 35, further comprising allowing communication of a group of data elements from the data word, between the sub-word paths, such that information may be shared between the sub-word paths.

37. A method, as in claim 36, wherein the information in said allowing communication is a data sub-word weight.

38. A method, as in claim 36, wherein the information in said allowing communication is at least one data element from the data sub-word.

39. A method, as in claim 36, wherein the information is at least one data element from a data word.

40. A method, as in claim 36, wherein said method further comprises inverting all data elements, within a data sub-word, whose significance is less than or equal to the significance of a particular data element within the data sub-word.

41. A method, as in claim 36, wherein said method further comprises inverting all encoded data elements, within an encoded data sub-word, whose significance is less than or equal to the significance of a particular encoded data element within the encoded data sub-word.

42. A method, as in claim 36, wherein said encoding encodes a data word by associating the data word with an encoded data word which is selected from a set of encoded data words where the encoded data word is of length z elements with a fixed number of data elements in each logic state.

43. A method, as in claim 36, wherein said encoding encodes a data sub-word by associating the data sub-word with the encoded data sub-word in a user defined way.

44. A method, as in claim 36, wherein the encoded data word is balanced.

45. A method, as in claim 36, wherein the encoded data word is substantially balanced.

46. A method, as in claim 36, further comprising using a code governor to change the weight of the encoded data word.

47. A method, as in claim 36, wherein said using a code governor is based at least in part on said allowing communication.

48. A method, as in claim 47, further comprising using a parity line to change the weight of the encoded data word.

49. A method, as in claim 48, wherein said using a parity line is based at least in part on said allowing communication.

50. A method, as in claim 36, wherein said encoding is a means for binomially encoding the data word.

51. A method, as in claim 50, further comprising adding 1 to the largest decimal number to be encoded.

52. A method, as in claim 50, wherein the encoded data word is balanced.

53. A method, as in claim 50, wherein the encoded data word is substantially balanced.

54. A method, as in claim 50, further comprising using a code governor to change the weight of the encoded data word.

55. A method, as in claim 54, wherein said using a code governor is based at least in part on said allowing communication.

56. A method, as in claim 54, further comprising using a parity line to change the weight of the encoded data word.

57. A method, as in claim 56, wherein said using a parity line is based at least in part on said allowing communication.

58. A decoder module to decode an encoded data word, whose encoded data elements occupy at least a first logic state and a second logic state, said decoder module comprising:

at least two sub-word paths, each of said at least two sub-word paths to receive an encoded data sub-word, comprising a set of data elements of the encoded data word and to allow communication, between said at least two sub-word paths, such that information is shared between said at least two sub-word paths; and

a decoder coupled with said at least two sub-word paths, said decoder to decode the encoded data sub-word into a data sub-word;

such that data sub-words form a data word.

59. A decoder as recited in claim 58, wherein said decoder inverts all data elements, within the data sub-word, whose significance is less than or equal to the significance of a particular data element within the data sub-word.

60. A decoder as recited in claim 58, wherein said decoder inverts all encoded data elements, within the encoded data sub-word, whose significance is less than or equal to the significance of a particular encoded data element within the encoded data sub-word.

61. A decoder as recited in claim 58, wherein the number of elements in the data sub-word is at least one less than the number of elements in the encoded data sub-word.

62. A decoder module, as in claim 58, further comprising decoding logic for decoding a value represented by the encoded data word that is not decoded by said decoder.

63. A decoder as recited in claim 58, wherein said decoder decodes the encoded data sub-word by associating the encoded data sub-word with the data sub-word in a user-defined way.

64. A decoder as recited in claim 58, wherein said decoder is a binomial decoder.

65. A decoder module, as in claim 58, wherein said decoder module is disposed on an integrated circuit die.

66. A method of decoding an encoded data word, whose encoded data elements occupy at least a first logic state and a second logic state, said method comprising:

receiving encoded data sub-words onto sub-word paths, the encoded data sub-words comprising sets of data elements of the encoded data word;

allowing communication between the sub-word paths, such that information may be shared between the sub-word paths; and

decoding the encoded data sub-words into data sub-words;

such that the data sub-words form a data word.

67. A method, as in claim 66, wherein the information, in said allowing communication, is an encoded data sub-word weight.

68. A method, as in claim 66, wherein the information, in said allowing communication, is at least one encoded data element from an encoded data sub-word.

69. A method, as in claim 66, wherein the information, in said allowing communication, is at least one encoded data element from the encoded data word.

70. A method, as in claim 66, wherein said method further comprises inverting all encoded data elements, within an encoded data sub-word, whose significance is less than or equal to the significance of a particular encoded data element within the encoded data sub-word.

71. A method, as in claim 66, wherein said method further comprises inverting all encoded data elements, within an encoded data sub-word, whose significance is less than or equal to the significance of a particular encoded data element within the encoded data sub-word.

72. A method, as in claim 66, wherein said decoding, decodes an encoded data sub-word by associating the encoded data sub-word with a data sub-word in a user defined way.

73. A method, as in claim 66, wherein said decoding is a means for binomially decoding a data sub-word.

74. A method, as in claim 73, further comprising adding 1 to a decimal value to be decoded.

75. A data processing system comprising:

at least two sub-word paths, each of said at least two sub-word paths to receive a data sub-word, comprising a set of data elements of a data word;

an encoder coupled with said at least two sub-word paths, said encoder to encode the data sub-word into an encoded data sub-word; such that encoded data sub-words form an encoded data word wherein an information content of the data word is spread between the encoded data sub-words and the weight of the encoded data sub-words.

a parallel encoded data line bus coupled with said at least two sub-word paths to receive the encoded data sub-words and to facilitate transmission of the encoded data sub-words;

at least two sub-word paths coupled with said parallel encoded data line bus, each of said at least two sub-word paths to receive an encoded data sub-word, comprising a set of data elements of the encoded data word and allowing communication, between said at least two sub-word paths, such that information is shared between said at least two sub-word paths; and

a decoder coupled with said at least two sub-word paths, said decoder to decode the encoded data sub-word into the data sub-word;

such that data sub-words form the data word.

76. A data processing system, as in claim 75, wherein said data processing system allows communication of a group of data elements from the data word, such that information is shared between said at least two sub-word paths.

77. A data processing system, as in claim 76, wherein the information is a data sub-word weight.

78. A data processing system, as in claim 76, wherein the information is at least one data element from the data sub-word.

79. A data processing system, as in claim 76, wherein the information is at least one data element from the data word.

80. An encoder as recited in claim 76, wherein said encoder inverts all the data elements, within the data sub-word, whose significance is less than or equal to the significance of a particular data element within the data sub-word.

81. An encoder as recited in claim 76, wherein said encoder inverts all encoded data elements, within the encoded data sub-word, whose significance is less than or equal to the significance of a particular encoded data element within the encoded data sub-word.

82. An encoder as recited in claim 76, wherein the number of elements in the encoded data sub-word is at least one greater than the number of elements in the data sub-word.

83. A data processing system, as in claim 76, further comprising encoding logic for encoding values represented by the data words that are not encoded by said encoder.

CLAIMS

84. A data processing system, as in claim 76, further comprising a code governor, wherein said code governor sets a logic state of at least one particular data element location in the encoded data word such that the number of data elements in each logic state is constant for each encoded data word.

85. A code governor, as recited in claim 84, wherein said code governor, uses in part, the information shared between said at least two sub-word paths to set the logic state of at least one particular data element in the encoded data word.

86. A data processing system, as in claim 84, further comprising a parity line to change the weight of the encoded data word by adding an element to the encoded data word.

87. A parity line, as recited in claim 86, wherein said parity line, uses in part, the information shared between said at least two sub-word paths to set the logic state of the element output onto said parity line.

88. An encoder as recited in claim 76, wherein said encoder encodes the data word by associating the data word with an encoded data word which is selected from a set of encoded data words where the encoded data word is of length z elements with a fixed number of data elements in each logic state.

89. An encoder as recited in claim 76, wherein said encoder encodes the data sub-word by associating the data sub-word with the encoded data sub-word in a user defined way.

90. An encoder as recited in claim 76, wherein said encoder is a binomial encoder.

91. A data processing system as recited in claim 76, wherein said encoder module is disposed on an integrated circuit die.

92. A data processing system, as in claim 76, further comprising a *Vdd* power rail wherein a sum of the current flowing in said *Vdd* power rail is substantially constant.

93. A data processing system, as in claim 76, further comprising:

a code governor; and

a *Vdd* power rail;

wherein said code governor sets the logic state of at least one particular data element location in the encoded data word, such that a sum of the current flowing in said *Vdd* power rail, is substantially constant.

94. A code governor, as recited in claim 93, wherein said code governor, uses in part, the information shared between said at least two sub-word paths to set a logic state of at least one particular data element in the encoded data word.

95. A data processing system, as in claim 93, further comprising a parity line to change the weight of the encoded data word by adding an element to the encoded data word.

96. A parity line, as recited in claim 95, wherein the information shared between the data sub-words is used, in part, to set a logic state of the element output onto said parity line.

97. A data processing system, as in claim 76, further comprising a *Vss* power rail wherein a sum of a current flowing in said *Vss* power rail is substantially constant.

98. A data processing system, as in claim 76, further comprising:

a code governor; and

a *Vss* power rail;

wherein said code governor sets the logic state of at least one particular data element location in the encoded data word, such that a sum of the current flowing in said *Vss* power rail is substantially constant.

99. A code governor, as recited in claim 98, wherein said code governor, uses in part, the information shared between said at least two sub-word paths to set a logic state of at least one particular data element in the encoded data word.]

100. A data processing system, as in claim 98, further comprising a parity line to change the weight of the encoded data word by adding an element to the encoded data word.

101. A parity line, as recited in claim 100, wherein the information shared between the data sub-words is used, in part, to set a logic state of the element output onto said parity line.

102. A data processing system, as in claim 76, further comprising encoded data lines wherein a sum of the current flowing in said parallel encoded data line bus is substantially constant.

103. A data processing system, as in claim 76, further comprising:
a code governor, wherein said code governor sets the logic state of at least one particular data element location in the encoded data word;
such that a sum of the current flowing in said parallel encoded data line bus is substantially constant.

104. A code governor, as recited in claim 103, wherein said code governor, uses in part, the information shared between said at least two sub-word paths to set a logic state of at least one particular data element in the encoded data word.

105. A data processing system, as in claim 103, further comprising a parity line to change the weight of the encoded data word by adding an element to the encoded data word.

106. A parity line, as recited in claim 105, wherein the information shared between the data sub-words is used, in part, to set a logic state of the element output onto said parity line.

107. A decoder as recited in claim 76, wherein said decoder inverts all data elements, within the data sub-word, whose significance is less than or equal to the significance of a particular data element within the data sub-word.

108. A decoder as recited in claim 76, wherein said decoder inverts all encoded data elements, within the encoded data sub-word, whose significance is less than or equal to the significance of a particular encoded data element within the encoded data sub-word.

109. A decoder as recited in claim 76, wherein the number of elements in the data sub-word is at least one less than the number of elements in the encoded data sub-word.

110. A data processing system, as in claim 76, further comprising decoding logic for decoding a value represented by the encoded data word that is not decoded by said decoder.

111. A decoder as recited in claim 76, wherein said decoder decodes the encoded data sub-word by associating the encoded data sub-word with the data sub-word in a user-defined way.

112. A decoder as recited in claim 76, wherein said decoder is a binomial decoder.

113. A data processing system, as in claim 76, wherein said data processing system is disposed on an integrated circuit die.

114. A method for transmitting a data word in a data processing system, said method comprising:

receiving data sub-words onto sub-word paths, the data sub-words comprising sets of data elements of the data word;

encoding the data sub-words into encoded data sub-words, such that the encoded data sub-words form an encoded data word wherein an information content of the data word is spread between the encoded data sub-words and the weight of the encoded data sub-words;

transmitting the encoded data sub-words over a parallel encoded data line bus;

receiving the encoded data sub-words onto the sub-word paths, the encoded data sub-words comprising sets of data elements of the encoded data word;

allowing communication, between the sub-word paths, such that information may be shared between the sub-word paths; and

decoding the encoded data sub-words into the data sub-words;

such that the data sub-words form the data word.

115 A method, as in claim 114, further comprising allowing communication of a group of data elements from the data word, between the sub-word paths, such that information may be shared between the sub-word paths.

116. A method, as in claim 115, wherein the information in said allowing communication is a data sub-word weight.

117. A method, as in claim 115, wherein the information in said allowing communication is at least one data element from the data sub-word.

118. A method, as in claim 115, wherein the information is at least one data element from a data word.

119. A method, as in claim 115, wherein said method further comprises inverting all data elements, within a data sub-word, whose significance is less than or equal to the significance of a particular data element within the data sub-word.

120. A method, as in claim 115, wherein said method further comprises inverting all encoded data elements, within an encoded data sub-word, whose significance is less than or equal to the significance of a particular encoded data element within the encoded data sub-word.

121. A method, as in claim 115, wherein said encoding encodes a data word by associating the data word with an encoded data word which is selected from a set of encoded data words where the encoded data word is of length **z** elements with a fixed number of data elements in each logic state.

122. A method, as in claim 115, wherein said encoding encodes a data sub-word by associating the data sub-word with the encoded data sub-word in a user defined way.

123. A method, as in claim 115, wherein the encoded data word is balanced.

124. A method, as in claim 115, wherein the encoded data word is substantially balanced.

125. A method, as in claim 115, further comprising using a code governor to change the weight of the encoded data word.

126. A method, as in claim 125, wherein said using a code governor is based at least in part on said allowing communication.

127. A method, as in claim 125, further comprising using a parity line to change the weight of the encoded data word.

128. A method, as in claim 127, wherein said using a parity line is based at least in part on said allowing communication.

129. A method, as in claim 115, wherein said encoding is a means for binomially encoding the data word.

130. A method, as in claim 129, further comprising adding 1 to the largest decimal number to be encoded

131. A method, as in claim 129, wherein the encoded data word is balanced.

132. A method, as in claim 129, wherein the encoded data word is substantially balanced

133. A method, as in claim 129, further comprising using a code governor to change the weight of the encoded data word.

134. A method, as in claim 133, wherein said using a code governor is based at least in part on said allowing communication.

135. A method, as in claim 133, further comprising using a parity line to change the weight of the encoded data word.

136. A method, as in claim 135, wherein said using a parity line is based at least in part on said allowing communication.

137. A method, as in claim 115, wherein the information, in said allowing communication, is an encoded data sub-word weight.

138. A method, as in claim 115, wherein the information, in said allowing communication, is at least one encoded data element from an encoded data sub-word.

139. A method, as in claim 115, wherein the information, in said allowing communication, is at least one encoded data element from the encoded data word.

140. A method, as in claim 115, wherein said method further comprises inverting all encoded data elements, within an encoded data sub-word, whose significance is less than or equal to the significance of a particular encoded data element within the encoded data sub-word.

141. A method, as in claim 115, wherein said method further comprises inverting all encoded data elements, within an encoded data sub-word, whose significance is less than or equal to the significance of a particular encoded data element within the encoded data sub-word.

142. A method, as in claim 115, wherein said decoding, decodes an encoded data sub-word by associating the encoded data sub-word with a data sub-word in a user defined way.

143. A method, as in claim 115, wherein said decoding is a means for binomially decoding a data sub-word.